05

Diferencias entre clases de excepciones

Transcripción

[00:00] Hemos visto ya que creamos nuestra propia excepción, un ArithmeticException. Está aquí, de hecho, en la memoria. Y no sucedió nada, aquí en el output de la consola vemos que solamente dio inicio, método main, método 1, 2, fin, fin, fin. Y no sucedió nada. Vamos a ver, por qué es que no sucedió nada.

[00:22] Vamos a partir desde el inicio, haciendo un repaso a lo que era la creación de objetos en la memoria HEAP. Bueno, la memoria HEAP, ya lo mencionamos en el primer curso de Java, es la memoria donde se van a almacenar todos los objetos que creamos en nuestro programa.

[00:40] Entonces aquí tenemos al ArithmeticException, que es el objeto que hemos creado y en método 2 hemos creado nuestra variable ae, que es la variable que hace referencia a este ArithmeticException que creamos nuevamente en la memoria HEAP que está aquí.

[00:56] ¿Pero nosotros qué necesitamos hacer con esa bomba, hacer con la ArithmeticException? Es que, por ejemplo, yo ya tengo la bomba aquí. ¿Qué necesito? Necesito detonarla. Entonces, lo que método 2 tiene que hacer aquí es lanzar fuera de método 2 esa bomba para que explote en la pila de ejecución. Entonces, con eso, él va a conseguir trabar la pila de ejecución.

[01:22] ¿Yo qué necesito? Lanzar la excepción, y recuerden, lanzar la excepción. Yo necesito lanzar la bomba fuera de método 2 para que consiga trabajar, digamos la ejecución de la pila, y si es que nadie sabe cómo tratarla, muere e

programa. Perfecto. Entonces vamos a verlo ya un poco más a nivel de código. ¿Como yo puedo lanzar una excepción? ¿Cómo creen?

[01:49] Nuevamente Java llega con una palabra reservada para este tipo de situaciones y es throw, que está aquí. ¿Esa palabra reservada qué significa? Lanza. Exacto. ¿Entonces, si yo quiero lanzar esa excepción qué hago? Throw ae, que es el nombre de mi excepción, que yo nombrado. Perfecto.

[02:21] Y ahora vemos que automáticamente mi código dejó de compilar, aquí en la última línea, línea 20, ya me está dando un error. ¿Por qué me da este error? Porque si yo estoy lanzando una excepción aquí, en la línea 19, o sea, si yo a propósito estoy mandando ese rollo, estoy diciendo, ¿sabes qué? Tú vas a explotar porque explotas, así de simple.

[02:42] No me importa si explotas porque se hizo una condición o no, tú explotas. Si yo estoy diciéndole esto en esta línea, por ende ninguna otra línea que está abajo del throw va a ejecutar su ciclo, entonces el throw es como la sentencia de muerte, tú explotas aquí. Entonces más abajo del throw no puede haber ninguna línea de código, porque simplemente no va a ser ejecutada, va a ser completamente ignorada, no tiene sentido tener aquí esta línea de código.

[03:10] El compilador nos dice eso, el compilador nos dice: "¿Sabes qué? Tú no puedes tener mas código abajo de un throw, así de simple." ¿Qué hago para eso? Lo borró. Y vamos a ver qué pasa. Hago clic, guardo y vemos que él inicia método 2 y ahora sí, en efecto, él lanzó mi bomba y lanzó mi ArithmeticException.

[03:34] Ahora, yo aquí declaré de una forma, digamos, un poco tradicional en la forma de que hablando ya de declaración de objetos, creación de objetos, esto es una forma tradicional de crear un objeto, pero cuando hablamos de lanzar excepciones, por lo general no se hace de esta forma. Lo que se hace es directamente lanzar el objeto, que es la excepción.

[03:53] No creo una referencia y después lanzo la referencia que apunta al objeto no. Lo que se hace tradicionalmente con el tema de lanzar excepciones es lanzar directamente la excepción. ¿Por qué? Porque es un objeto, es un objeto del tipo excepción que yo estoy lanzando directamente.

[04:12] No necesito referenciarlo a alguna variable porque él va a ser lanzado directamente a la pila de ejecución. Perfecto. Entonces ahora vemos que yo puedo lanzar nuevos objetos, yo puedo lanzar una excepción y ahora me entra la curiosidad. ¿Será que si por ejemplo vamos a crear aquí otra clase?

[04:36] Vamos a llamarla cuenta solamente para fines ilustrativos, y la voy a dejar vacía, no quiero una clase cuenta como la anterior. Vamos a guardar aquí, no quiero una clase cuenta como la anterior, simplemente quiero un objeto cualquiera. ¿Será que yo consigo hacer declarar aquí, por ejemplo cuenta cuenta igual new Cuenta?

[05:02] Perfecto, cree una referencia a este objeto cuenta que yo he creado aquí. ¿Y será que yo consigo throw cuenta? A ver. Vemos que no me deja. ¿Por qué yo no puedo hacer un throw de cuenta si también es un objeto y la excepción también es un objeto? La explicación sencilla es que solamente podemos hacer throw de objetos, que son excepciones, esa es la regla de Java.

[05:34] Tú solamente puedes lanzar excepciones. Errores, es lo único que puedes lanzar. Si tú tienes digamos objetos tipo cuenta, cliente, objetos de negocio o cualquier objeto que no sea una excepción, tú no puedes usar throw, porque tú no puedes lanzar cualquier objeto a la memoria y esperar que tu código colapse.

[05:55] Esa es la la premisa básica. Entonces, ahora ya sabemos lanzar nuestro propio ArithmeticException. Él va a lanzar a la memoria y va a detener todo el ciclo de ejecución. Y llega una nueva pregunta. ¿Será que de la misma forma en la que Java crea NullPointerException, ArithmeticException, etcétera nosotros conseguimos de hecho crear nuestras excepciones personalizadas?

[06:25] Por ejemplo, no hay saldo excepción o alguna cosa así. ¿Será que conseguimos hacerlo? Vamos a ver eso en la siguiente clase. Nos vemos.